A. Problem Statement:
To design an automated system that automatically takes attendance and makes a corresponding excel sheet that contains the detail. This system should be able to capture photos to analyse or add new student and also should be able to train this image on the fly. The system should be able to take attendance of each student under 5 sec.

B. objectives:
The main objective of this project is to develop a facial recognition app. It is built with Python, OpenCV and Tkinter which has reliability for application performance. This application which offers good user interface and high reliability.
C. System architecture:

There are three major modules in the project:
1. Capturing Images
2. Training Images
3. Tracking Images

1. CAPTURING IMAGES
This module is provided for new users to add their image using OpenCV along with corresponding Roll no and Name of the student. A total of 30 images are taken and written into training images folder.

2. TRAINING IMAGES:
This module uses the images that were captured during capturing images module and extract facial feature using haar cascading and embed them into Trainner.yml file this file is used by Tracking Image module to check if the person is recognizable or not.

3. TRACKING IMAGES
This module is used to recognize the person using yml file created during training module and if the person is recognized the it automatically created a excel file and add the attendance automatically to it along with a message in the main tinker program.

IV. DESIGN:
After gathering the requirement analysis, the next step is to design to model or the system. This phase is critical for any project as the user-interface should meet the customer's demand and also, it forms the base for the coding or the implementation phase. If the design is not proper and according to the user's requirements, then the task of the developers becomes more challenging. Designing not only shows how the user-interfaces appears but also helps in understanding how the system should work. It defines the functionality of each module that has to be implemented. The developer can understand the system more clearly and hence, his task becomes easy. So, this holds a lot of importance over the other phases. We design various types of diagrams that help us understand the system better.
A. Sequence Diagram:

UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram. Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications.

B. Use Case Diagram:

A use case diagram is a graph of actors set of use cases enclosed by a system boundary, communication associations between the actors and users and generalization among use cases. The use case model defines the outside(actors) and inside (use case) of the system’s behaviour. Use case diagram is quite simple in nature and depicts two types of elements: one representing the business roles and the other representing the business processes.

c. State Chart Diagram:

The state graph contains the game-plan of states, occasions and exercise. This graph noteworthy for tending to the lead of the interface, class and made effort. The key centralization of state outline is to show the occasion sort out lead of the request.

V. Result and Discussion:

The working of the system we proposed involves three steps: 1. Creating a database.
2. Training the model.
3. Recognizing the real-time data. Before starting, let us import all necessary libraries. The libraries includes,
1. Opencv: to process the image/video.
2. Numpy: to work with pixels in form of arrays.
3. Onnx: to work with onnx models.
5. Os: to change/create directory and also to open files in the directory.
6. Imutiles: to manipulate the image (preprocessing).
7. Tensorflow: to create layers and create/load models.
8. Pickle: to serialize or deserialize to python objects (like list, tuples, dictionary).
9. Xlsxwriter: to create/read/write to excel sheet.
VI. CONCLUSION:

In order to obtain the attendance of individuals and to record their time of entry and exit, the authors proposed the attendance management system based on face recognition technology in the institutions/organizations. The system takes attendance of each student by continuous observation at the entry and exit points. The result of our preliminary experiment shows improved performance in the estimation of the attendance compared to the traditional black and white attendance systems. Current work is focused on the face detection algorithms from images or video frames.

In further work, authors intend to improve face recognition effectiveness by using the interaction among our system, the users and the administrators. On the other hand, our system can be used in a completely new dimension of face recognition application, mobile based face recognition, which can be an aid for common people to know about any person being photographed by cell phone camera including proper authorization for accessing a centralized database.

VII. REFERENCES

[1] Taiping Zhang, Yuan Yan Tang, Bin Fang, Zhaowei Shang, Xiaoyu L